

REMARKS

The Office Action dated April 19, 2004 has been received and carefully noted. The above amendments to the claims and the following remarks are submitted as a full and complete response thereto.

Claims 1-19 have previously been cancelled without prejudice or disclaimer of the subject matter recited therein, claims 20-43 are pending in the present application. Claims 20, 32, and 43 are independent claims. Claims 20, 32, and 43 have been amended to more particularly point out and distinctly claim the present invention. Support for the subject matter added to claims 20, 32, and 43 may generally be found throughout the specification of the present application and, more specifically, at least on page 5, lines 3-26, from page 8, line 14, through page 9, line 4, and on page 7, lines 14-29.. No new matter has been added. Claims 20-43 are respectfully submitted for consideration.

Rejection of Claim 10 Under 35 U.S.C. § 103(a):

Claims 20-43 have been rejected under 35 U.S.C. § 103 (a) as being unpatentable over *Software Engineering: A Practitioner's Approach* by an unspecified author (hereafter "*Software Engineering*") in view of *System Analysis & Design with Modern Mehtods* by Fertuck (hereafter "*System Analysis*"). It is acknowledged in the Office Action that *Software Engineering* fails to disclose that normalization is a way to achieve high cohesion and low coupling. However, it is alleged in the Office Action that *Software Engineering* may be combined with *System Analysis* to yield the subject matter

recited in claims 20-43. Applicant respectfully submits that claims 20-43 recite subject matter which is neither disclosed nor suggested by *Software Engineering and System Analysis*, taken either individually or in combination.

Claim 20, upon which claims 21-31 depend, recites a method of implementing an application and of eliminating uncontrolled internal interdependencies within the application. The application recited in claim 20 includes a number of functional entities, wherein each entity includes one or more elements, and the application produces application output data from input data such that element output data produced by the elements determines entity output data produced by the functional entities and the entity output data determines the application output data. As recited in claim 20, there are interdependencies formed between the elements, between the functional entities, or between the elements and the functional entities. The method includes normalizing an element such that uncontrolled internal interdependencies within the element are eliminated. The normalizing step includes dividing the application input data into parts small enough for each element input data to solely determine the corresponding element output data and analyzing interdependencies between each entity input data and the corresponding entity output data. The normalizing step also includes performing a search for such combinations of entity input data which solely determine the entity output data and forming a normalized element out of the found combinations, such that the element input data solely determines the element output data.

Claim 32, upon which claims 33-42 depend, recites a system for implementing an application and eliminating uncontrolled internal interdependencies within the application. The application includes a number of functional entities, each entity including one or more elements, and the application produces application output data from input data such that element output data produced by the elements determines entity output data produced by the functional entities and the entity output data determines the application output data. As recited in claim 32, there are interdependencies formed between the elements, between the functional entities, or between the elements and the functional entities. The system recited in claim 32 includes normalizing means for normalizing one or more elements such that uncontrolled internal interdependencies within each element are eliminated. The normalizing means includes dividing means for dividing the application input data into parts small enough for each element input data to solely determine the corresponding element output data and analyzing means for analyzing interdependencies between each entity input data and the corresponding entity output data. The normalizing means also includes performing means for performing a search for such combinations of entity input data which solely determine the entity output data and forming means for forming a normalized element out of the found combinations, such that the element input data solely determines the element output data.

Claim 43, recites a system for implementing an application and eliminating uncontrolled internal interdependencies within the application. The application includes a number of functional assemblies, each entity including one or more elements, and the

application produces application output data from input data such that element output data produced by the elements determines assembly output data produced by the functional assemblies and the assembly output data determines the application output data. As recited in claim 43, there are interdependencies formed in at least one of between the elements, between the functional assemblies, or between the elements and the functional assemblies. The system recited in claim 43 includes the one or more elements. The system also includes normalizing means for normalizing one or more elements such that uncontrolled internal interdependencies within each element are eliminated. The normalizing means recited in claim 43 includes dividing means for dividing the application input data into parts small enough for each element input data to solely determine the corresponding element output data and analyzing means for analyzing interdependencies between each entity input data and the corresponding entity output data. The normalizing means also includes performing means for performing a search for such combinations of entity input data which solely determine the entity output data and forming means for forming a normalized element out of the found combinations, such that the element input data solely determines the element output data.

The methods and systems for implementing an application and eliminating uncontrolled internal interdependencies within the application recited in claims 20-43 of the present application allow for significant improvements over the prior art. For example, the methods and systems according to the claimed invention allow for easier control of interdependencies between parts contained in an application. Also, such

methods and systems allow for the elimination of uncontrolled internal interdependencies between parts within a technical application. It is respectfully submitted that *Software Engineering* and *System Analysis*, taken either individually or in combination, fail to disclose or suggest the elements of any of the presently pending claims. Therefore, it is respectfully further submitted that *Software Engineering* and *System Analysis*, taken either individually or in combination, fail to provide at least the above-discussed advantages of the claimed invention.

Software Engineering discloses that a “program architecture [may be] manipulated according to a set of heuristics (guidelines)” (page 361, first paragraph of §13.6). *Software Engineering* also discloses evaluating “the ‘first iteration’ of the program structure to reduce coupling and improve cohesion”, evaluating “module interfaces to reduce complexity and redundancy and improve consistency” and defining “modules whose function is predictable, but avoiding modules that are overly restrictive” (page 361, subsections I and IV-V of §13.6). *Software Engineering* further discloses striving “for ‘controlled entry’ modules, avoiding ‘pathological connections’” (page 363, subsection VI of §13.6).

However, *Software Engineering* fails to disclose or suggest at least “normalizing an element such that uncontrolled internal interdependencies within said element are eliminated,” as recited in claims 20-31 of the present application, wherein the normalizing step includes “dividing the application input data into parts small enough for each element input data to solely determine the corresponding element output data,

analyzing interdependencies between each entity input data and the corresponding entity output data, performing a search for such combinations of entity input data which solely determine the entity output data, and forming a normalized element out of the found combinations, such that the element input data solely determines the element output data”. In addition, *Software Engineering* also fails to disclose or suggest at least “normalizing means for normalizing one or more elements such that uncontrolled internal interdependencies within each element are eliminated,” as recited in claims 32-43 of the present application, where the normalizing means include “dividing means for dividing the application input data into parts small enough for each element input data to solely determine the corresponding element output data, analyzing means for analyzing interdependencies between each entity input data and the corresponding entity output data, performing means for performing a search for such combinations of entity input data which solely determine the entity output data, and forming means for forming a normalized element out of the found combinations, such that the element input data solely determines the element output data”.

System Analysis discloses “the design of ‘hard-core’ applications that are difficult to prototype because they do not involve much user interaction” (page 434, first paragraph). *System Analysis* also discloses that “[n]ormalization is a way to achieve high cohesion and low coupling in database design” and that “[c]onversion to a higher normal form increases cohesion by eliminating attributes that are not about the same entity, as defined by the same key” (page 464, fourth paragraph).

However, like *Software Engineering, System Analysis* also fails to disclose or suggest at least “normalizing an element such that uncontrolled internal interdependencies within said element are eliminated,” as recited in claims 20-31 of the present application, wherein the normalizing step includes “dividing the application input data into parts small enough for each element input data to solely determine the corresponding element output data, analyzing interdependencies between each entity input data and the corresponding entity output data, performing a search for such combinations of entity input data which solely determine the entity output data, and forming a normalized element out of the found combinations, such that the element input data solely determines the element output data”. In addition, like *Software Engineering, System Analysis* also fails to disclose or suggest at least “normalizing means for normalizing one or more elements such that uncontrolled internal interdependencies within each element are eliminated,” as recited in claims 32-43 of the present application, where the normalizing means include “dividing means for dividing the application input data into parts small enough for each element input data to solely determine the corresponding element output data, analyzing means for analyzing interdependencies between each entity input data and the corresponding entity output data, performing means for performing a search for such combinations of entity input data which solely determine the entity output data, and forming means for forming a normalized element out of the found combinations, such that the element input data solely determines the element output data”. In other words,

System Analysis fails to address or eliminate any of the above-discussed shortcomings of *Software Engineering*.

Applicant respectfully points out that *Software Engineering* and *System Analysis* each disclose ‘normalizing’ procedures that are non-analogous to the normalizing steps and means recited in the claimed invention. Rather, *Software Engineering* and *System Analysis* each disclose procedures for the normalization of databases that, at best, are analogous only to the procedures disclosed in the portion of the present specification related to the prior art. More specifically, neither *Software Engineering* nor *System Analysis* discloses or suggests any of the “dividing”, “analyzing”, “performing”, and “forming” steps and means recited in claims 20-43.

At least in view of the above claim amendments and remarks, reconsideration and withdrawal of the rejection of claims 20-43 under 35 U.S.C. 103(a) over *Software Engineering* in view of *System Analysis* is respectfully requested.

Applicant respectfully submits that all of the comments included in the Office Action have been addressed and that all of the rejections included therein have been overcome. Hence, Applicant respectfully further submits that, at least in view of the above, claims 20-43 of the present application contain allowable subject matter. Therefore, it is respectfully requested that all claims pending in the present application be allowed, and that this application be passed to issue.

If for any reason the Examiner determines that the application is not now in condition for allowance, it is respectfully requested that the Examiner contact, by

telephone, the Applicant's undersigned representative at the indicated telephone number to arrange for an interview to expedite the disposition of this application.

In the event this paper is not being timely filed, Applicant respectfully petitions for an appropriate extension of time. Any fees for such an extension together with any additional fees may be charged to Counsel's Deposit Account 50-2222.

Respectfully submitted,

A handwritten signature in black ink, appearing to read 'Hermes M. Soyez', written over a horizontal line.

Hermes M. Soyez, Ph.D.
Registration No. 45,852

Customer No. 32294
SQUIRE, SANDERS & DEMPSEY LLP
14TH Floor
8000 Towers Crescent Drive
Tysons Corner, Virginia 22182-2700
Telephone: 703-720-7800
Fax: 703-720-7802
HMS